# Arvados - Story #7241

## [Keep] Prototype Azure blob storage

09/08/2015 06:49 PM - Brett Smith

| | | | |
|---|---|---|---|
| **Status:** | Resolved | **Start date:** | 09/23/2015 |
| **Priority:** | Normal | **Due date:** | |
| **Assigned To:** | Tom Clegg | **% Done:** | 100% |
| **Category:** | Keep | **Estimated time:** | 0.00 hour |
| **Target version:** | 2015-09-30 sprint | | |

**Description**

The prototype should implement the Keep volume interface using Azure blob storage, including returning errors that are required to report problems in the underlying storage.

The prototype does not need to deal with non-essential errors like configuration problems, temporary network hiccups, etc.

Ideally the prototype will be developed in such a way there's a clear path for further development can make it production-ready. However, in case of doubt or conflict, getting the prototype done in a timely manner to prove the concept overrides this concern.

The branch review should ensure that the prototype meets functionality requirements, and can meet known scalability requirements in production use. It doesn't need to address code style, issues with tests (although ideas for tests are good to capture), etc.

Refs

- https://msdn.microsoft.com/en-us/library/azure/dd179377.aspx
- https://godoc.org/github.com/Azure/azure-sdk-for-go/storage

**Subtasks:**

| | |
|---|---|
| Task # 7371: Write minimal AzureBlobVolume | **Resolved** |
| Task # 7373: Test with a real container | **Resolved** |
| Task # 7372: Stub Azure services for tests | **Resolved** |
| Task # 7379: Deal with Delete races | **Resolved** |
| Task # 7380: Add install docs or wiki page | **Closed** |

**Related issues:**

| | | |
|---|---|---|
| Blocks Arvados - Feature #7159: [Keep] Implement an Azure blob storage volume... | **Resolved** | **08/28/2015** |

### Associated revisions

**Revision 2c07efe6 - 10/02/2015 08:06 PM - Tom Clegg**

Merge branch '7241-azure-blob-volume' closes #7241

### History

**#1 - 09/08/2015 08:24 PM - Brett Smith**

*- Description updated*

*- Category set to Keep*

**#2 - 09/15/2015 06:32 PM - Brett Smith**

*- Story points set to 5.0*

Arbitrarily assigning five because we expect one engineer to do this and no other stories for a sprint.

**#3 - 09/15/2015 07:30 PM - Brett Smith**

*- Assigned To set to Tom Clegg*

**#4 - 09/23/2015 01:28 PM - Tom Clegg**

*- Status changed from New to In Progress*

**#5 - 09/24/2015 07:10 PM - Tom Clegg**

Progress so far, at da74a60:

- GET, PUT, and index work against real Azure service.

TODO:

- Test with multiple volumes & volume types (e.g., make sure flag parsing isn't broken).
- Make generic volume tests pass (implement the mock Azure service).
- Heed readonly flag.
- Handle races between Delete and Touch/Put as required by Volume interface spec.
- Handle races between concurrent Puts. (Specifically make sure "create blob + die before committing" doesn't delete a blob that was just written here by another goroutine/process.)
- Write setup/usage notes on wiki. Using Keep with Azure Storage?

#### #6 - 09/24/2015 08:19 PM - Tom Clegg

*- Description updated*

#### #7 - 09/25/2015 03:18 PM - Peter Amstutz

- azureVolumeAdder.Set() needs to check that azureStorageAccountName was provided.
- Under what circumstances is io.ErrUnexpectedEOF acceptable?  Presumably that occurs on a short read.  Is the assumption that we don't know if the block is bad or not, so it will be caught further up the stack when it checks the MD5 sum?
- I did some digging into the distinction between "Block Blobs", "Page blobs" and "Append blobs":
  - It looks like a "block blob" is similar to how we define a file in a Keep manifest; the blob is chunked into a list of N blocks up to 4MiB in size, where individual blocks are identified by "blockid".
  - "Page blobs" are basically block devices (in the Unix sense) that are addressed by offset.  When creating a new page blob we can supply up to 1 TiB of data, subsequent updates can be up to 4MiB at a time.
  - "Append blobs" start out empty and accept 4 MiB writes added to the end.
- After a careful reading of the REST API I think there's a better way to create the block blob using a single PUT API call instead of 3 API calls: https://msdn.microsoft.com/en-us/library/azure/dd179451.aspx however it doesn't look like it is supported by the Go SDK!
- There's an official stub service for blob store testing: https://azure.microsoft.com/en-us/documentation/articles/storage-use-emulator/ but I assume it only runs on Windows
- A better way to implement Touch() might be to use Set Blob Properties https://msdn.microsoft.com/en-us/library/azure/ee691966.aspx with x-ms-sequence-number-action: increment
- The DELETE method https://msdn.microsoft.com/en-us/library/azure/dd179413.aspx supports conditional headers https://msdn.microsoft.com/en-us/library/azure/dd179371.aspx to solve the race between Put/Touch/Delete

#### #8 - 09/25/2015 04:00 PM - Peter Amstutz

Using PutBlock() (instead of PUT blob) we can't write more than 4 MiB at a time:
(this is uploading a 37 MiB file as a block:)

```
2015/09/25 11:46:44 azure-storage-container:"keepstore": Write(f63fd24143c2ce19b20f40dbe956d2f1): storage: ser
vice returned error: StatusCode=413, ErrorCode=RequestBodyTooLarge, ErrorMessage=The request body is too large
 and exceeds the maximum permissible limit.
RequestId:819fa3a4-0001-0025-2ea9-f7b04e000000
Time:2015-09-25T15:44:55.8301168Z, RequestId=819fa3a4-0001-0025-2ea9-f7b04e000000
2015/09/25 11:46:44 [[::1]:36587] PUT f63fd24143c2ce19b20f40dbe956d2f1 0.646792s 500 5 "Fail"
```

#### #9 - 09/25/2015 07:00 PM - Tom Clegg

Peter Amstutz wrote:

> - Under what circumstances is io.ErrUnexpectedEOF acceptable?  Presumably that occurs on a short read.  Is the assumption that we don't know if the block is bad or not, so it will be caught further up the stack when it checks the MD5 sum?

Yes. ErrUnexpectedEOF is ReadFull's way of saying it reached EOF before reading len(buf) bytes, which is normal in this context because len(buf) is always BlockSize. Without doing an additional API call (or improving the SDK to show us the Content-Length header) we don't know the data size at this point, so we just read until EOF and rely on the MD5. (It's also possible the HTTP library itself notices that the body is shorter than advertised by Content-Length, but even if it doesn't, I think this is reliable with just the MD5 check.)

(Other changes forthcoming. +1 adding PutBlob to the Azure SDK. The 3-step process is slightly icky already but gets way worse (especially in the Azure service stub) if we have to store a variable number of blocks per blob.)

#### #10 - 09/29/2015 07:07 PM - Tom Clegg

Peter Amstutz wrote:

> - azureVolumeAdder.Set() needs to check that azureStorageAccountName was provided.

Fixed.

> - After a careful reading of the REST API I think there's a better way to create the block blob using a single PUT API call instead of 3 API calls: https://msdn.microsoft.com/en-us/library/azure/dd179451.aspx however it doesn't look like it is supported by the Go SDK!

Indeed. The 4 MiB block limit makes the three-step workaround even worse. Added the atomic "create and write up to 64 MiB" SDK method in https://github.com/Azure/azure-sdk-for-go/pull/218.

> - There's an official stub service for blob store testing: https://azure.microsoft.com/en-us/documentation/articles/storage-use-emulator/ but I assume it only runs on Windows

"The storage emulator uses a local Microsoft SQL Server instance and the local file system..."

FWIW: The Azure Go SDK's test suite uses the real Azure service (you have to give it credentials in env vars).

> - A better way to implement Touch() might be to use Set Blob Properties https://msdn.microsoft.com/en-us/library/azure/ee691966.aspx with x-ms-sequence-number-action: increment

Unfortunately the "sequence number" stuff is only for page blobs.

Changed this to use SetBlobMetadata (https://github.com/Azure/azure-sdk-for-go/pull/219), which bumps Last-Modified and Etag.

> - The DELETE method https://msdn.microsoft.com/en-us/library/azure/dd179413.aspx supports conditional headers https://msdn.microsoft.com/en-us/library/azure/dd179371.aspx to solve the race between Put/Touch/Delete

So they say. I fought it for a bit but couldn't get it to obey If-Unmodified-Since -- but If-Match works, and Etag does change when you update metadata (as we do in Touch). It involves an extra API call, but it seems safe.

Setting the headers required a third SDK change. There's an open issue #209 for this, and the expected implementation is still up in the air, so I've done something expedient in our fork in the meantime. When their issue 209 is done (by us or someone else) we can use that instead and drop our custom code.

#### #11 - 09/30/2015 07:00 PM - Peter Amstutz

Non-blocker comments:

- Instead of the "-azure-storage-replication" option, it's possible to query the storage account https://msdn.microsoft.com/en-us/library/azure/ee460802.aspx and determine the number of replications (3 or 6) from AccountType.

- When you tried If-Unmodified-Since, how did you representing timestamps?  This describes the format that is accepted: https://msdn.microsoft.com/en-us/library/azure/dd135714.aspx

#### #12 - 09/30/2015 07:10 PM - Tom Clegg

Peter Amstutz wrote:

> - Instead of the "-azure-storage-replication" option, it's possible to query the storage account https://msdn.microsoft.com/en-us/library/azure/ee460802.aspx and determine the number of replications (3 or 6) from AccountType.

Yes, that sounds like a better way to get a default, at least.

> - When you tried If-Unmodified-Since, how did you representing timestamps?  This describes the format that is accepted: https://msdn.microsoft.com/en-us/library/azure/dd135714.aspx

Used RFC1123. You can see exactly what I tried here

https://github.com/curoverse/azure-sdk-for-go/blob/master/storage/blob_test.go#L318-L335

#### #13 - 10/02/2015 08:35 PM - Tom Clegg

*- Status changed from In Progress to Resolved*

*- % Done changed from 83 to 100*

Applied in changeset arvados|commit:2c07efe6ac7455059f2fccd558ea796f9c315e19.